HIERARCHICAL STOCHASTIC ACTIVITY NETWORKS

MOHAMMAD ABDOLLAHI AZGOMI AND ALI MOVAGHAR

Department of computer engineering,
Sharif university of technology,
Tehran 11365, Iran.

E-mail: *azgomi@mehr.sharif.edu* *and* *movaghar@sharif.edu*

Abstract: *Stochastic activity networks* (SANs) are a powerful and flexible extension of Petri nets. These models can be used for the modeling and analysis of various kinds and different aspects of distributed real-time systems. Similar to other classical extensions of Petri nets, SANs have some limitations for modeling complex and large-scale systems. In order to remove these limitations and provide some high-level modeling facilities, we have defined a new extension for SANs, which is called *hierarchical stochastic activity networks* (HSANs). HSAN models provide facilities for composing a hierarchy of submodels, incremental modeling and the reusability of submodels. HSAN models encapsulate hierarchies and a key benefit of these models is the possibility of automatic employment of composition techniques by their modeling tool. In this paper, we present the graphical notations, formal definition, and methods for the analysis of HSAN models. We also present an example of the application of these models in the design of a high-capacity packet-based telecommunication switch.

keywords: Stochastic Petri Nets, Stochastic Activity Networks, High-Level Petri Nets, Hierarchical Modeling.

1. INTRODUCTION

Stochastic activity networks (SANs) [Movaghar and Meyer, 1984] are a stochastic generalization of Petri nets (PNs). These models are more powerful and flexible than most other stochastic extensions of Petri nets such as *stochastic Petri nets* (SPNs) [Molloy, 1982] and *generalized stochastic Petri nets* (GSPNs) [Marsan, Balbo and Conte 1986]. SANs permit the representation of concurrency, timeliness, fault-tolerance and degradable performance in a single model [Movaghar, 1985].

Similar to other classical extensions of Petri nets, SANs have some limitations for modeling complex and large-scale systems, such as the lack of compositionality, incremental modeling and the reusability of submodels. In order to remove these limitations, we have defined a new extension for SANs called *hierarchical stochastic activity networks* (HSANs). HSAN models encapsulate hierarchies and a key benefit of these models is the possibility of automatic employment of composition techniques, such as Replicate/Join [Sanders and Meyer, 1991] or Graph Composition [Stillman, 1999] formalisms, by their modeling tools.

In this paper, we present properties, graphical notation, formal definition, and applications of HSANs. The paper is organized as follows. In Sec. 2, some limitations of SANs and related works on hierarchical PNs are presented. In Sec. 3, the informal and formal definitions and graphical notations of HSANs are presented. Methods for the transformation and analysis of HSANs will be introduced in Sec. 4. And, in Sec. 5, an example of the application of these models is presented.

2. MOTIVATION

2.1 Limitations of SANs

SAN models have been employed in a lot of applications and are supported with a few powerful modeling tools such as *UltraSAN* [Sanders et al, 1995] and Mobius [Deavours et al, 2002]. But, both the ordinary definition [Movaghar and Meyer, 1984] and the new definition of SANs [Movaghar, 2001] have the following limitations:

1. *SANs are rather flat:* People manage complexity by having hierarchies. Programming languages, especially object-oriented languages, have simple, intuitive ways of encapsulating hierarchies [Deavours, 2003]. The Replicate/Join formalism has been proposed to alleviate this problem. Unfortunately, There is a set of issues with this construct [Deavours, 2003]. It is limited to a tree-like structure, and an arbitrary symmetric model structure, such as ring or mesh, cannot be expressed with this construct. However, the use of the Replicate/Join or the Graph Composition formalisms is specific to modeling tools and not SANs in general.
2. *The lack of facilities for compositionality*: There is no facility in the definition of SANs for constructing models for complex systems with small and simple submodels.
3. *The lack of facilities for reusability:* Since SAN models are flat, using a part of an existing model as a component for constructing a new one is difficult.
4. *The lack of facilities for incremental modeling:* There is a lack of facilities for constructing

Proceedings 17th European Simulation Multiconference
(c) SCS Europe BVBA, 2003 ISBN 3-936150-25-7

complex models incrementally, by starting with abstract components and easily replacing them with detailed and enhanced components.

2.2 Related Works

There are a few numbers of extensions for PNs, which provide facilities for composing hierarchical models. Some of them are as follows:

- *Hierarchical Petri nets (HPNs):* The absence of compositionality has been one of the main critiques raised against Petri net models [Jensen, 1990]. Because, PN models are a flat network of *places* and *transitions*. To remove this drawback, *hierarchical Petri nets (HPNs)* were introduced. Hierarchical Petri nets are a structural method of describing concurrent processes. They enable to design more complex systems through abstracting some parts of the net.
- *Hierarchical coloured Petri nets (HCPNs)*: Hierarchical extensions have also been introduced for coloured Petri nets (CPNs) [Jensen, 1990]. *Hierarchical coloured Petri nets (HCPNs)* provide facilities for constructing a hierarchy of CPNs [Huber, Jensen and Shapiro, 1990]. In HCPNs, CPN models can be structured into *pages* (subnets) using *substitution transitions*. HCPNs also offer the concept of *place fusion*. *Fusion places* make it possible to specify that a set of places represent a single conceptual place, which has been drawn as several copies. When tokens are removed or added to a fusion place all places in the fusion set will have their marking correspondingly changed [Lakos, 1995].
- *Modular coloured Petri nets (MCPNs):* Allows invariant analysis in the context of modular nets incorporating both place and *transition fusion* [Christensen and Petrucci, 1992]. MCPNs therefore extend hierarchical coloured Petri nets (HCPNs) by supporting the notion of *substitution places* as well as *substitution transitions*. These constructs are also referred to as *super places* and *super transitions*, respectively.
- *Replicate/Join construct* [Sanders and Meyer, 1991]: This construct has been introduced to provide model composition in the *UltraSAN* modeling tool. Replicate and Join operations are used to compose models by sharing states. *Subnet* of SAN model is the unit of composition. These subnets can be composed in a hierarchical manner to construct a SAN model. This approach has been extended and generalized in the work of [Stillman, 1999] on the *Graph Composition* formalism in the Mobius modeling framework [Deavours, 2001].

3. Definitions of HSANs

To remove some limitations of SANs for modeling and analysis of complex and large-scale systems, we introduced *hierarchical stochastic activity networks* (HSANs). These models provide constructs to build a hierarchy of submodels rather than viewing SANs as a flat collection of primitives. The structure of the composition in HSANs is a hierarchy of submodels. Each level of the hierarchy represents a different level of abstraction. This will facilitate the comprehension of the whole model of a large system.

3.1 The Elements of HSAN Models

An HSAN model is composed of five primitives of the ordinary SANs, (including, *place*, *input gate*, *output gate*, *instantaneous activity*, *timed activity*) and *macro activity* (MA). MA is an HSAN submodel, which is composed of a finite number of SAN primitives and lower-level macro activities.

Place fusion provides a mechanism for interfacing macro activities to other parts of an HSAN model. A MA may have zero or more *input* and *output fusion places*. Fusion places are a subset of normal places. A MA has a well-defined interface, which is similar to high-level programming languages. The places surrounding a MA are *formal fusion places*. When a MA is used in an HSAN model, these formal places will be bound by *actual places,* which are normal places of SANs. This relation is similar to formal parameters of a procedure, which will be bound by actual parameters of the caller program in high-level programming languages. A formal fusion place has always the same marking as the related actual place. The two places are different views of the same place.

3.2 Graphical Notation

In graphical representation of HSANs, a fusion place is depicted as ◎. A graphical representation of a MA has been depicted in Fig. 1. In this figure, a MA is drawn as a rectangle, which is surrounded by input and output fusion places. An example of an HSAN model is presented in Fig. 2 and the definition of *TMA1* in Fig. 3.

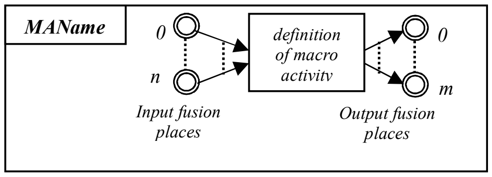
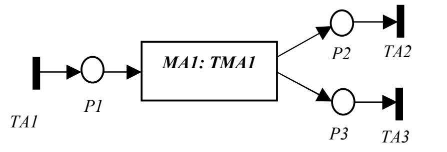


Fig. 1. Definition of macro activity

Fig. 2. An example of the usage of a *MA* named *TMA1*



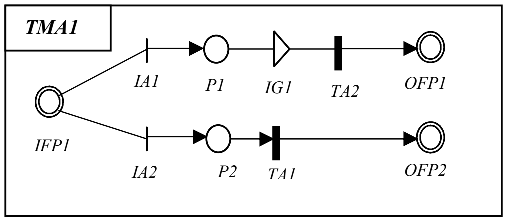


Fig. 3. Definition of a macro activity named *TMA1*

3.3 Some Rules and Properties

There are the following rules and properties on HSAN models:

1. In order to interface with other models or submodels, a MA should have at least one input fusion place or one output fusion place. The maximum number of input or output fusion places is not limited.
2. Inside a MA, all input fusion places are *read-only* and all output fusion places are *write-only*.
3. All predicates and functions of a MA are dependent to the marking of the local or fusion places.
4. A MA can be used in the hierarchy of an HSAN model more than once and with different names.
5. In each level of the hierarchy of an HSAN model, naming is local. For example, in Fig. 2, there are three timed activities *TA1*, *TA2* and *TA3*. In Fig. 3, which is the definition of *TMA1*, there are other timed activities named *TA1* and *TA2*. However, in the same level (root HSAN or in each MA), names should be unique.
6. MAs can be defined separately. Previously defined MAs may be used to compose a new HSAN model.

3.4 Formal Definition of HSANs

Formal definition of HSANs is the basis for analytic solution, simulation, and for the formal verification over state spaces. Based on these definitions and well-defined syntax and semantics, HSAN tools will facilitate the process of constructing HSAN models. Modeling tools for HSAN will automatically check for the correct use of these syntax and semantics.

Now, we formally define *hierarchical stochastic activity networks (HSANs)*, based on a new definition of SANs [Movaghar, 2001]. In the following definitions, N denotes the set of natural numbers and R_+ represents the set of non-negative real numbers.

Definition 1. *Hierarchical stochastic activity network (HSAN)* is defined as a 12-tuple $HSAN = (P, IA, TA, MA, IG, OG, IR, OR, C, F, \Pi, \rho)$ where:

- P is a finite set of *places*,
- IA is a finite set of *instantaneous activities*,
- TA is a finite set of *timed activities,*
- MA is a finite set of *macro activities*, which will be defined later,
- IG is a finite set of *input gates*. Each input gate has a finite number of inputs. To each $G \in IG$, with m inputs, is associated a *function* $f_G : N^m \to N^m$, called the function of G, and a predicate $g_G : N^m \to \{true, false\}$, called the *enabling predicate* of G,
- OG is a finite set of *output gates*. Each output gate has a finite number of outputs. To each $G \in OG$, with m outputs, is associated a *function* $f_G : N^m \to N^m$, called the function of G,
- $IR \subseteq P \times \{1, \ldots, |P|\} \times IG \times (IA \cup TA \cup MA)$ is the *input relation*. IR satisfies the following conditions:
  - for any $(P_1, i, G, a) \in IR$ such that G has m inputs, $i \le m$,
  - for any $G \in IG$ with m inputs and $i \in N$, $i \le m$, there exist $a \in (IA \cup TA \cup MA)$ and $P_1 \in P$ such that $(P_1, i, G, a) \in IR$,
  - for any $(P_1, i, G_1, a), (P_1, j, G_2, a) \in IR$, $i = j$ and $G_1 = G_2$,
- $OR \subseteq (IA \cup TA \cup MA) \times OG \times \{1, \ldots, |P|\} \times P$ is the *output relation*. OR satisfies the following conditions:
  - for any $(a, i, G, P_1) \in OR$ such that G has m outputs, $i \le m$,
  - for any $G \in OG$ with m outputs and $i \in N$, $i \le m$, there exist $a \in (IA \cup TA \cup MA)$ and $P_1 \in P$ such that $(a, G, i, P_1) \in OR$,
  - for any $(a, G_1, i, P_1), (a, G_2, j, P_1) \in OR$, $i = j$ and $G_1 = G_2$,
- $C : N^n \times IA \to [0, 1]$ is the *case probability function*, where $n = |P|$.
- $F = \{F(.|\mu, a); \mu \in N^n, a \in TA\}$ is the set of *activity time distribution functions*, where $n = |P|$ and, for any $\mu \in N^n$, and $a \in TA$, $F(.|\mu, a)$ is a probability distribution function,
- $\Pi : N^n \times TA \to \{true, false\}$ is the *reactivation predicate*, where n is defined as before,
- $\rho : N^n \times TA \to R_+$ is the *enabling rate function*, where n is defined as before.

Definition 2. *Macro activity (MA)* is defined as a 14-tuple $MA = (P, IA, TA, MA, IG, OG, IR, OR, C, F, \Pi, \rho, IFP, OFP)$ where:

- P, IA, TA, MA, IG, OG, IR, OR, C, F, Π, and ρ are defined as before,
- $IFP \subseteq P$ is a set of *input fusion places*,
- $OFP \subseteq P$ is a set of *output fusion places*,
- $|IFP \cup OFP| > 0$.

4. ANALYSIS OF HSAN MODELS

An HSAN model is solvable by analytic or simulative methods, *iff* the dependency graph of the HSAN model is *acyclic*. In this graph, nodes are macro activities and the composition relation between MAs and their parents determines arcs. The root of this graph is the HSAN model.

If the dependency graph is acyclic, it is possible to employ an algorithm to transform an HSAN model to an equivalent flat SAN model. Also, it is possible to automatically employ methods for the analysis of composed models. Some of these techniques are described in this section.

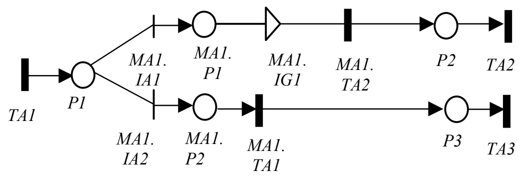
4.1 Transformation to a Flat HSAN Model

Each HSAN model has a behaviorally equivalent SAN model. For the analysis of an HSAN model, it is possible to transform it to a behaviorally equivalent SAN model. A step-by-step substitution algorithm does transformation. In each step, MAs on leaves of the graph will be substitute by their implementations. This will be repeated until the only node on the graph is the root node.

As we mentioned before, naming in HSAN models is local. To resolve the problem of duplicate names in substitution, the name of each element of a MA will be preceded by the name of its parent MA. For example, in transformation of HSAN model of Fig. 2, which is shown in Fig. 4, *IA1, IA2, P1, P2, IG1, IG1, TA1, TA2* are preceded by *MA1* and a dot. Also, all marking dependent predicates and functions of MAs should be substituted.

The resulting model is a flat SAN model, without any MA. If the resulting model is Markovian, the reachability graph for this model can be generated and methods for solution of Markov chains can be used to solve it. If the model is not Markovian, the simulation method may be employed.

Fig. 4. An equivalent SAN model for the model of Fig. 2.



4.2 Using Replicate/Join Construct

The main disadvantage of the ad-hoc transformation of HSAN models to flat SANs is the explosion of state-space. There are a few techniques for constructing SAN models in a way, which avoid state-space explosion problem. One of these techniques is the Replicate/Join construct [Sanders and Meyer, 1991], which we briefly described in Sec. 2.2.

A key benefit of HSAN models is the possibility of automatic employment of composition formalisms by modeling tools. Therefore, a possible way of the analysis of HSAN models is their automatic transformation to a Replicate/Join construct.

For this purpose, the modeling tool can check for a tree-like structure in the dependency graph of an HSAN model. After that, it can automatically find shared states based on the fusion places and organize the model using Replicate and Join operations. For the solution of the resulting model, the technique proposed in [Sanders and Meyer, 1991] can be employed.

The model of a telecom switch, which is presented in the next section, is an example of such kind of HSAN models.

4.3 Using Graph Composition Formalism

Another composition formalism, which has been proposed in the Mobius modeling framework for SANs and other models, is the Graph Composition formalism [Stillman, 1999]. This formalism does not limit the model hierarchy to a tree-like structure and any arbitrary is possible.

A modeling tool for HSANs can also check for the possibility of the use of this formalism. Then, it can automatically transform the HSAN model to this construct and then use its method of solution, which is proposed in [Stillman, 1999].

4.4 Simulation of HSAN Models

If an HSAN model or one of its macro activities includes non-exponential timed activities or its state-space is infinite, it may not be solved analytically. In such cases, discrete-event simulation may be employed to solve the model.

For the efficient simulation of HSAN models, methods proposed for SANs, such as procedures presented in [Sanders and Freire, 1993], can be used. These methods are used with Replicate/Join formalism. Therefore, after the transformation of HSAN model to this construct, these methods of simulation may be employed.

If an HSAN model is transformed to the Graph Composition formalism, the methods of simulation proposed in the Mobius modeling framework can be used.

5. AN HSAN MODEL FOR A PACKET-BASED TELECOM SWITCH

In this section, we present an HSAN model for a high-capacity telecommunication switch (HCTS). In contrast to traditional class 5 TDM switches, HCTS is a packet-based switch based on gigabit Ethernet technology. In this switch, analog voice signals will be converted to voice packets in line-cards and will be switched by Ethernet switches to output ports. We have constructed a model for this switch in three levels: *card level*, *unit level*, and *rack level*. The purpose of this modeling is to show the application of HSANs in this area.

HCTS has a modular structure, including one *control rack* (CR) and a few numbers of *subscriber/trunk racks* (STRs). CR includes a *control unit (CU)* and a few numbers of *subscriber units* (SUs) or *trunk units* (TUs). In CU, there is a two modules redundancy (TMR) of *control and switching module card* (CSMC), which executes *call-processor* program of the switch. An HCTS can operate if at least one of these two modules is working properly. STRs include a few number of SU or TU. In this example, we concentrate on modeling SUs, which include line-cards. The goal of this modeling is to evaluate some design parameters, which are related to the *quality of service* (QoS) of the switch to the analog/digital subscribers. Three important parameters in our study are the *blocking probability*, the *maximum delay of voice packets in switching system* (from line-cards to output ports), and the *optimal size of input buffers* of switches in different parts of the system.

Each STR includes 6 SUs. The connection of these units to the whole system is through a gigabit Ethernet switch multiplexer. Each SU includes 19 *line-cards* (LCs). Star topology is used to connect LCs to an Ethernet switch multiplexer with 100 *Mbps* input ports and 1 *Gbps* output port. Each LC includes 30 subscriber ports. LC packetizes 4 *ms* analog voice signals to 84 bytes packet. Output multiplexer of LC is a 100 *Mbps* switch.

A simplified HSAN model for an HCTS is presented in Fig. 5. This model is composed of *n* rack macro activities (*rma)*, input buffer places (*ibs*), and Ethernet switching-hub macro activity (*eshma*). In Fig. 6, *rma* is depicted, which is composed of 6 unit macro activities (*uma*), *ibs*, *esma* and the output buffer fusion place (*ob*). In Fig. 7, *uma* is depicted, which is composed of 19 line-card macro activities (*lcma*), *ibs*, Ethernet switch macro activity (*esma*) and *ob*. In Fig. 8, *lcma* is depicted. An *lcma* is composed of 30 subscriber port macro activities (*spma*), *ibs*, *esma* and *ob*. In Fig. 9, *spma* is depicted. For this purpose, *off-hook* timed activity models the process of hooking off a phone set. *release* timed activity models the holding time. *busy* is a place, which presents the busy state of line. *isbusy* is an input gate for *packetize* timed activity. *isbusy* checks whether the line is busy or not. *packetize* models the conversion of 4 *ms* analog voice signals to 84 bytes packets by digital signal processor (DSP) of LC. *ob* is an output fusion place of *spma*. Finally, in Fig. 10, *esma* is a simple model for an Ethernet switch multiplexer with *n* input ports and a single output port. The model for an Ethernet switching-hub (*eshma*) is similar to *esma.*

Fig. 5. HSAN model of an HCTS (top level)

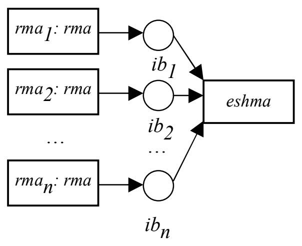


Fig. 6. Rack macro activity (*rma*)

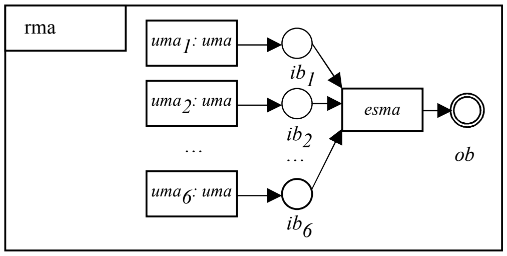


Fig. 7. Unit macro activity (*uma*)

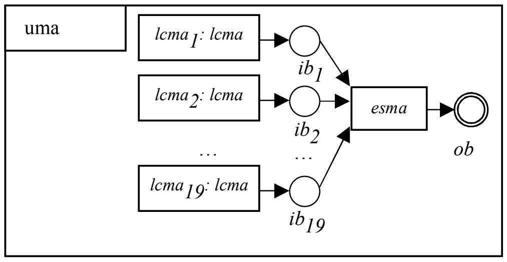


Fig. 8. Line-card macro activity (*lcma*)

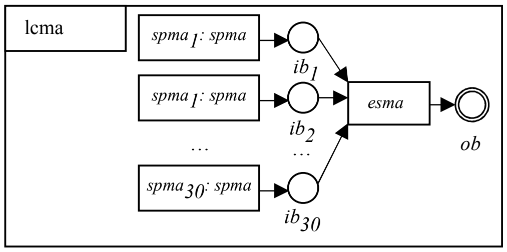
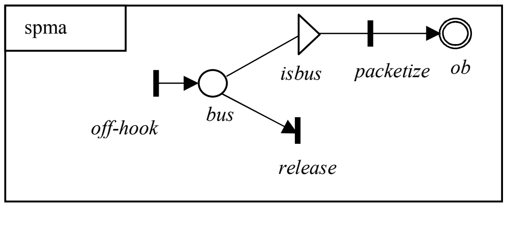


Fig. 9. Subscriber port macro activity (*spma*)



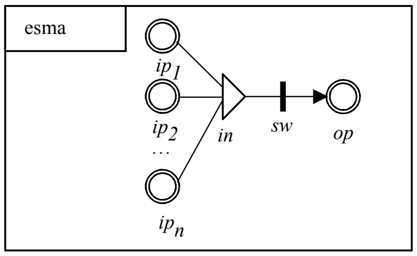


Fig. 10. Ethernet switch macro activity (*esma*)

6. TOOL SUPPORT

For modeling and analysis with HSANs, we have developed a modeling tool called *SANBuilder*. This tool is an enhanced and completely redesigned version of our existing tool for SANs called *SharifSAN* [Abdollahi and Movaghar, 2001]. *SANBuilder* enables modelers to construct HSAN models in a graphical editor, define and store separate macro activities and retrieve them to reuse in a new HSAN model. It also provides features for interactive simulation (i.e. token-game animation), automatic simulation, and analytic solution of HSAN models.

7. CONCLUDING REMARKS

In this paper, we introduced *hierarchical stochastic activity networks* (HSANs). HSAN models provide facilities for composing a hierarchy of submodels, incremental modeling and the reusability of submodels. HSANs have the same power as ordinary SANs, but HSAN models are more flexible and high-level. HSANs have formal definition and well-defined syntax and semantics. HSAN models encapsulate hierarchies and a key benefit of these models is the possibility of automatic employment of composition techniques, such as Replicate/Join or Graph Composition formalisms.

Similar to ordinary SANs, HSANs can be used for the modeling and analysis of various kinds and different aspects of computer and communication systems. HSANs can be used to build both Markovian and non-Markovian models, and have nondeterministic, probabilistic, and stochastic settings. For functional analysis (i.e., verification), the nondeterministic setting of HSAN models and a method for specifying the properties of system (e.g. *temporal logic*) may be employed. For the analysis of operational aspects (i.e., performance, dependability or performability evaluation), the stochastic setting of these models and methods for their steady state or transient analytic solution or simulation can be used.

We have developed *SANBuilder* tool for the modeling and analysis of HSAN models. To make this tool more useful for the application on large-scale systems, we are working on methods for efficient solution and simulation of HSAN models.

REFERENCES

Abdollahi Azgomi M. and Movaghar A. 2001, "*SharifSAN*: A Tool for Verification and Performance Evaluation Based on a New Definition of SANs," In *Proc. of the 13th IASTED Int. Conf. on Parallel and Distributed Computing and Systems (PDCS'01)*, Anaheim, CA, Pp667-672.

Christensen S. and Petrucci L. 1992, "Towards a Modular Analysis of Coloured Petri Nets," In *Proc. of Int. Conf. on Application and Theory of Petri Nets (LNCS 616)*, K. Jensen (ed.), Springer-Verlag, Pp113-33.

Deavours D.D. 2001, *Formal Specification of The Mobius Modeling Framework*, Ph.D. Dissertation, University of Illinois at Urbana-Champaign.

Deavours D.D. 2003, Personal Correspondence, Apr 2003.

Deavours, D.D. et al 2002, "The Mobius Framework and Its Implementation," *IEEE Trans. on Soft. Eng.*, Vol. 28, No. 10, Pp956-969.

Huber P., Jensen K. and Shapiro R.M. 1990, "Hierarchies of Coloured Petri Nets," In *Proc. of 10th Int. Conf. on Application and Theory of Petri Nets (LNCS 483)*, Springer-Verlag, Pp313-341.

Jensen K. 1990, *Coloured Petri Nets: A High Level Language for System Design and Analysis (LNCS 483)*, Springer-Verlag.

Lakos C.A. 1995, "From Coloured Petri Nets to Object Petri Nets," In *Proc. of 16th Int. Conf. on the Application and Theory of Petri Nets (LNCS 935)*, Torino, Italy, Springer-Verlag, Pp278-297.

Marsan M.A., Balbo G. and Conte G. 1986, *Performance Models of Multiprocessor Systems*, MIT Press.

Molloy M.K. 1982, "Performance Analysis Using Stochastic Petri Nets," *IEEE Trans. on Computers*, C-31, Pp913-917.

Movaghar A. and Meyer J.F. 1984, "Performability Modeling with Stochastic Activity Networks," In *Proc. of the 1984 Real-Time Systems Symp.*, Austin, TX, USA, Pp215-224.

Movaghar A. 1985, *Performability Modeling with Stochastic Activity Networks*, Ph.D. Dissertation, University of Michigan.

Movaghar A. 2001, "Stochastic Activity Networks: A New Definition and Some Properties," *Scientia Iranica*, Vol. 8, No. 4, Pp303-311.

Sanders W.H. and Freire R.S. 1993, "Efficient Simulation of Hierarchical Stochastic Activity Network Models," *Discrete Event Dynamic Systems: Theory and App.*, Vol. 3, no. 2/3, Pp271-300.

Sanders W.H. and Meyer J.F. 1991, "Reduced Base Model Construction Methods for Stochastic Activity Networks," IEEE J. on Selected Areas in Comm., Special Issue on Computer-Aided Modeling, Analysis, and Design of Comm. Net. , Vol. 9, No. 1, Pp25–36.

Sanders W.H. et al 1995, "The UltraSAN Modeling Environment," *Performance Evaluation*, Vol. 24, Pp1-33.

Stillman A.J. 1999, *Model Composition in the Mobius Modeling Framework*, M.S. Thesis, University of Illinois at Urbana-Champaign.